P =? NP - Problem

Marcel Schiwarth
Marcel Herrlich

29. Juni 2009

Motivation

Einführung

- Bisherige Betrachtungen: Algorithmen mit polynomialer Laufzeit
- Algorithmen mit nicht- bzw. superpolynomialer Zeit.
- Unlösbare Probleme wie Turingsche Halteproblem
- Unterscheidung zw. Optimierungsalgorithmen vs. Entscheidungsalgorithmen
- Wichtiges Anwendungsgebiet: Verschlüsselung insb. RSA

Gliederung

Übersicht - Problemklassen

- Klasse $\mathcal{P}$
  - Leichte Probleme, Bsp SHORTEST-PATH
  - Aufwand $\mathcal{O}(n^k)$
- Klasse $\mathcal{EXPTIME}$
  - schwere Probleme, Bsp. LONGEST-PATH
  - mit $\mathcal{O}(k^n)$
  - es gilt $\mathcal{P} \subset \mathcal{EXPTIME}$
- Klasse $\mathcal{NP}$
  - Lösung mittels Nichtdeterminismus
  - unklar ob $\mathcal{NP} \subset \mathcal{EXPTIME}$,
  - oder $\mathcal{P} = \mathcal{NP}$

Komplexitätsklasse P

Beschreibung

- Menge konkreter Entscheidungen
- in polynomialer Zeit lösbar,$\mathcal{O}(n^k)$, für k konstant

Definition:

Die Funktion f mit $f : \{0,1\}^* \rightarrow \{0,1\}^*$, ist in polynomialer Zeit berechenbar, wenn ein Algorithmus A mit $\mathcal{O}(n^k)$ existiert, der für eine Eingabe $x \in \{0,1\}^*$ die Ausgabe $f(x)$ erzeugt.

Komplexitätsklasse P (Forts.)

PATH-Problem: Entscheidungsproblem

Algorithmus prüft ob G einen ungerichteten Graphen darstellt, $u, \ v \ \epsilon \ in \ G$, Benutzung der Breitensuche für die Berechnung des Kürzestesten Pfades v. u nach v in G, vergleicht Anzahl der Knoten auf dem erhaltenen Pfad mit k

- Halteproblem nur Akzeptanzproblem aber kein Entscheidungsproblem

Verifikationalgorithmus

Naive Methode:

- Auflisten aller Permutationen, der Knoten aus G
- jede Permutation wird überprüft ob der
- Pfad hamiltonisch ist
- m ... Anzahl der Knoten, n länge der Kanten
- m! sind die Anzahl der möglichen Permutationen je Knoten
- Laufzeit $O(m!) = \mathcal{O}(\sqrt{n!}) = \sqrt{2\pi} \cdot (n/e)^n$

Verifikationsalgorithmus Forts.

A(x, y),
x: Eingabestring,
y: als Zertifikat bezeichneter binärer String,
A besitzt zwei Argumente

Es gilt:

A verifiziert x, wenn ein y $\in (0, 1)^*$ existiert, sodass $A(x, y) = 1$
d. h. das Zertifikat y muss existieren.
$L = \{x \in \{0, 1\}^*\}$

NP

Definiton:

Die Klasse der **NP**-Probleme ist die Menge aller Probleme, die nichtdterministisch in **P**olynomzeit gelöst werden können.

- nichtberechenbare Probleme liegen nicht in $\mathcal{NP}$
- vermutlich existieren berechenbare Probleme die ebenfalls nicht in $\mathcal{NP}$ liegen
- jedes deterministisch in exponentieller Zeit lösbare Problem $\in \mathcal{NP} \Rightarrow$ nicht bekannt

poly. Reduktion

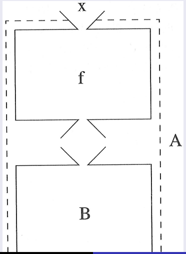
Definition

Seien A und B Sprachen (Probleme) mit A $\subseteq \Sigma_1^*$ und B $\subseteq \Sigma_2^*$. Dann heißt A polynomial reduzierbar auf B, geschrieben: A $\leq_p$ B, wenn es eine totale und mit polynomialer Komplexität berechenbare Funktion $f : \Sigma_1^* \rightarrow \Sigma_2^*$ gibt, sodass für alle $x \in \Sigma_1^*$ gilt: $x \in A \Leftrightarrow f(x) \in$ B.

- spezielle Form der Reduktion
- gefordert, dass Reduktion deterministisch in Polynomzeit berechnet werden kann

Beispiel

- unbekanntes Problem B der Klasse $\mathcal{NP}$
- bekanntes und ähnliches $\mathcal{NP}$ Problem A (z.B. SAT-Problem) notwendig
- Reduktion von Problem A auf Problem B
- damit kann bewiesen werden dass Problem B $\mathcal{NP}$-vollständig ist



Regeln

1. Falls A $\leq_p$ B und B $\in \mathcal{P}$, dann gilt A $\in \mathcal{P}$
2. Falls A $\leq_p$ B und B $\in \mathcal{NP}$, dann gilt A $\in \mathcal{NP}$
3. Falls A $\leq_p$ B und A $\notin \mathcal{P}$, dann gilt B $\notin \mathcal{P}$

NP-schwer

Definition:

Ein Problem (eine Sprache) A heißt $\mathcal{NP}$-schwer, wenn für alle Probleme (Sprachen) L $\in \mathcal{NP}$ gilt: L $\leq_p$ A.

- nicht notwendigerweise in $\mathcal{NP}$
- zum Beweise $\mathcal{NP}$-schweres Problem notwendig
- $\rightarrow$ Transitivität von $\leq_p$ kann genutzt werden, da L $\leq_p$ Y für alle L $\in \mathcal{NP}$ gilt
- $\Rightarrow$ es gilt: L $\leq_p$ A für alle L $\in \mathcal{NP}$
- $\Rightarrow$ entspricht genau unserer Definition, somit ist A $\mathcal{NP}$-schwer

NPC

Definition:

A heißt $\mathcal{NP}$-vollständig,wenn A $\mathcal{NP}$-schwer ist und A $\in \mathcal{NP}$ gilt.

Vermutung

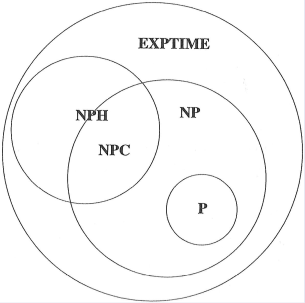


Abbildung: gilt unter der Annahme $\mathcal{P} \neq \mathcal{NP}$

Problem

Frage

- nicht bekannt ob $\mathcal{NP} = \mathcal{EXPTIME}$ und $\mathcal{P} = \mathcal{NP}$ gelten
- da $\mathcal{P} \neq \mathcal{EXPTIME}$ gilt, muss eine der Aussagen gelten
    - $\mathcal{P} \subset \mathcal{NP}$
    - $\mathcal{NP} \subset \mathcal{EXPTIME}$
    - $\mathcal{P} \subset \mathcal{NP} \subset \mathcal{EXPTIME}$
- bis heute keine Polynomzeit-Lösung für ein $\mathcal{NP}$-Probleme gefunden

Konsequenz

Satz

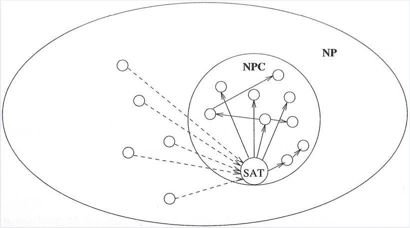
Wenn A ein $\mathcal{NPC}$ Problem ist, dann gilt:
$A \in \mathcal{P} \Leftrightarrow \mathcal{P} = \mathcal{NP}$

- das dieser Satz gilt muss für ein einziges $\mathcal{NPC}$-Problem gezeigt werden
- dann gilt es für **alle** Probleme
- polynomiale Transformation würde Verwendung finden, um einen Algorithmus für Problem x zu erhalten

NP-Nachweis

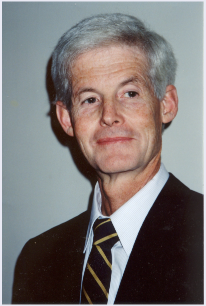
Vorgehensweise

1. Zeigen, dass B $\in \mathcal{NP}$ gilt
2. Wählen eines -Problems A, dass B ähnelt
3. überführen der Eingaben für A in Eingaben für B mittels polynomieller Transformation
4. Zeigen, dass $x \in A \Leftrightarrow f(x) \in$ B gilt



Wer war S.A.Cook?

- geboren am 14.12.1939 in Buffalo, NY
- Prof. der Informatik an der University of Toronto in Kanada
- Hauptbetätigungsfeld: Komplexitätstheorie
- berühmt durch Satz von Cook
- 1982 bekam er für diese Entdeckung den Turing-Award
- Website an der Universität http://www.cs.toronto.edu/ sacook/



SAT-Problem

Problem

- SAT $\in \mathcal{NP}$
- gesucht: Belegung der Variablen x_i aus w, f das en Ausdruck wahr wird
- z.B. $(\neg x_1 \vee x_3 \vee x_5) \wedge (\neg x_1 \vee x_2 \vee \neg x_4) \wedge (x_1 \vee x_2 \vee \neg x_3)$
- insgesamt 2^5 Möglichkeiten zur Belegung der Variablen, somit exponentieller Aufwand
- durch Orakel und Verifikation eines geratenen 5 Tupel $\rightarrow$ polynomialer Gesamtaufwand

Verschlüsselung

- Praktisch Unlösbare Probleme finden sich insbesondere in der Verschlüsselung wieder
- Besonders interessant ist hier der Entwurf dieser Probleme
- unterschieden wird Kryptographie, -logie und -analyse
- Praxis: Nicht oder nur mit erheblichen Aufwand Verfahren zwecks Sicherheit

Verschlüsselungsverfahren

- Schlüssel zur Codierung von Daten
- Schlüssel wird vom Empfänger der codierten Daten benötigt
- Wie soll der Schlüssel übertragen werden? Verschlüsselt oder Unverschlüsselt?!

Idee: öffentlicher Schlüssel RSA

RSA

- Erfinder: Rivest, Shamir, Adleman
- Öffentlicher Schlüssel $\mathcal{C}$ bestehend aus zwei Teilen e, q
- Nachricht mit $\mathcal{C}$ von beliebigen Absender verschlüsselt
- Entschlüsselung unter Verwendung des private Schlüssels $\mathcal{D}$ bestehend aus d , q
- d wird aus e und q ermittelt, wobei q sich aus den Primzahlen p1, p2 zusammensetzt

Abschließende Betrachtungen

- $\mathcal{NP}$-Vollständigkeits-Probleme finden in der Praxis großes Interesse
- z. B. Routenplanung
- Keine effizienten Algorithmen vorhanden
- Lösung mittels Näherungsalgorithmen z. B. Heuristiken- NÄCHSTE GRUPPE!
- Das $\mathcal{P} =? = \mathcal{NP}$-Problem gehört zu den Millennium-Problemen, für dessen Lösung 1 Millionen Dollar angesetzt sind

Ende

Fragen zur Thematik?

Vielen Dank für Ihre Aufmerksamkeit!